COMPLIANCE COMPONENT

Last Updated: 04/19/2005

DEFINITION	
Name	Data Element Definition Standards
Description	Increasing data creation, production, storage and availability does not increase the States' ability to use, share and make sense out of that data (Data Consumption). Information must be of high value to encourage confidence between interacting agencies. The challenge of obtaining Data Integrity requires proper design, processes that match the business requirements, good communication skills and constant vigilance. Data Definitions within a DBMS are rules that can be applied to table columns to enforce different types of data integrity. The surest way of enforcing relational integrity is to apply constraints at the server. These constraints will provide quality assurance to all data that enter the database, regardless of the data's source. However, some purchased applications enforce integrity on the "client side," using the application software to enforce the relationships. This provides much greater opportunity for violations of the database's relational integrity. Not only could parts of the application fail to adequately check all of the incoming data, but also data which is loaded "via the backdoor" (such as during migration of legacy data or feeds from other systems) could easily compromise the integrity of the database.
Rationale	The IT industry is migrating from a universe where system architecture and infrastructure are the commanding principles to a universe where application success is directly tied to content, context and value. Poor data quality creates artificial barriers to the effective use of information, leading to lost opportunities and decreased performance. In most cases, the cost associated with poor data quality are not only ignored but subsumed into the overall category of the cost of doing business. Yet, real costs are associated with nonconforming data – and they add up. The State of Missouri can reduce these significant costs by instituting a data quality improvement program as a core component of its business intelligence strategy. Multiple agencies wishing to share information must be able to describe what the information "looks like" so that when data arrives at its target location, the receiving agency can actually do something with it. A data standard provides the guidelines through which interacting agencies can confidently exchange information. The goal of a data standard is to enable the sharing or exchange of information between multiple agencies in a way that guarantees that the interacting agencies share the same understanding of what is represented within that information. When exchanged information is comprised of structured data, a data standard provides the description of that structure. A data standard, at the very least, defines entity names, data element names, descriptions, definitions and formatting rules. In addition, a data standard may include procedures, implementation guidelines and usage directives. As more information is being exchanged in different operating environments, the need for defined data standards is becoming more

	acute. Particularly in environments where many separate organizations (each with its own data definition peculiarities) have agreed to exchange data, there is a need to coordinate that information exchange in a way that provides the most benefit to all participants. Lack of Data Element Definition Standards increases risk: • Investment risk can be jeopardized as a result of poor data quality, either in infrastructure or effort. • Legal and regulatory risk due to external legal or regulatory boards exercising some control over products developed or processes executed within the organization (ex. Sarbanes-Oxley Act). Bad data can result in noncompliance with defined laws and regulations and lead to a serious risk. • Professional Risk resulting from a failure of a project, tagged with those associated with the project, scarring the states' reputation.
Benefits	The most important concept to keep in mind when discussing the definition and use of data element standards is predictability. Predictability associated with information exchange allows application developers to design and implement application architectures that can exploit expectations to allow for more efficient processing. Therefore, the expected benefits of predictability include: 1. Enabling effective sharing of information between collaborating partners - improving communication that, in turn, improves collections. 2. Reducing the amount of manual intervention in information processing and facilitating straight-through processing, which increases productivity and can reduce costs. 3. Providing a means for publishing the data element standards for the benefit of information exchange partners. 4. Streamlining access to improve knowledge-worker workflow. 5. Improving the quality, consistency, and interoperability of enterprise information. 6. Supporting the ongoing adoption of the use of standard data elements in coordination with any kind of application or system modernization. 7. Promoting the migration to a services-based architecture, which will simplify the process for improving and extending production systems. By properly defining data, the DBMS will ensure that the data adheres to a predefined set of rules. This alleviates the applications from enforcing some of the data integrity, which is prone to more errors and spreads the business rules to multiple locations, making consistency and modifications difficult. Enforcing integrity constraints at the DBMS level: • provides a single point of integrity enforcement, • reduces the reliance on each application to enforce the integrity, and • increases consistency by providing a single point of documentation for the integrity enforcement. Defining and enforcing Data Element Definition Standards will significantly increase the states' data quality. Since knowledge stems from data, this would increase the states' knowledge quality as well.

ASSOCIATED ARCHITECTURE LEVELS	
Specify the Domain Name	Information
Specify the Discipline Name	Data Management
Specify the Technology Area Name	Enterprise Data Element Standards

Specify the Product Component Name	
COMPLIANCE COMPONENT TYPE	
Document the Compliance Component Type	Standard
Component Sub-type	
COMPLIANCE DETAIL	
State the Guideline, Standard or Legislation	**1. Data Domain (a.k.a. Data Type)** Data type and length are the most fundamental integrity constraints applied to data in a database. Simply by specifying the data type for each column when a table is created, the DBMS automatically ensures that only the correct type of data is stored in that column. Processes that attempt to insert or update the data to a non-conforming value will be rejected. Furthermore, a maximum length is assigned to the column to prohibit larger values from being stored in the table. The data type and length of each column must be chosen wisely. It is almost always best to choose the data type that most closely matches the domain of correct values for the column. This is known as "Strong Data Typing". In general, adhere to the following rules: • If the data is numeric, favor SMALLINT, INTEGER, or DECIMAL data types. FLOAT is also an option for very large numbers only. • If the data is character, use CHAR or VARCHAR data types. • If the data is date and time, use DATE, TIME, and TIMESTAMP data types. • If the data is multimedia, use GRAPHIC, VARGRAPHIC, BLOB, CLOB, or DBCLOB data types. The benefits of using the proper data types are many, including: • ensuring data integrity because the DBMS will ensure that only valid data values are stored • the ability to use data arithmetic • a vast array of built-in functions to operate on and transform data values • multiple formatting choices • computers process numeric data more efficiently • character data requires more physical storage than numeric data (which is propagated to Indexes, Foreign Keys, backups, etc.), therefore requiring more I/O • DBMS Optimizers, being mathematical in nature, perform best on numeric data types It is better in terms of integrity and efficiency to store the data based on its domain. Users and programmers shall format the data for display instead of forcing the data into display mode for storage in the database. If data formatting is required of the DBMS, then User-Defined Functions and/or Views shall be used. **2. Null Rule** A null is a rule defined on a single column that allows or disallows inserts or updates of rows containing a null (the absence of a value) in that column. A null value is essentially the absence of a value (no value), although there are different kinds of null values. If business requirements for null value specifications are to isolate the difference between a legitimate null value and a missing value, then a separate column should be defined adding these deeper characterizations for the data when defined as null:

1. No value - there is no value for this field - a true null.
2. Unavailable - there is a value for this field, but for some reason it has been omitted. Using the unavailable characterization implies that at some point the value will be available and the field should be completed.
3. Not applicable - this indicates that in this instance, there is no applicable value.
4. Not classified - there is a value for this field, but it does not conform to a predefined set of domain values for that field.
5. Unknown - the fact that there is a value is established, but that value is not known.

Any null value specification must include a second column specifying the kind of null as an assigned representation. Different kinds of representations along with their related meanings include:

Representation meaning
"-" "no value"
"U" "unknown"
"X" "unavailable"
"N" "not applicable"

If the business requires to know why the value is missing, then a separate column needs to be defined that stores the type of NULL when the column is NULL.

The presence of spaces or “blanks” in the place of nulls represents a threat to referential integrity.

3. Unique Column Values
A unique value defined on a column (or set of columns) allows the insert or update of a row only if it contains a unique value in that column (or set of columns).

4. Primary Key Values
A primary key value defined on a key (a column or set of columns) specifies that each row in the table can be uniquely identified by the values in the key.

5. Referential Integrity Rules
A rule defined on a key (a column or set of columns) in one table that guarantees that the values in that key match the values in a key in a related table (the referenced value).

Referential integrity also includes the rules that dictate what types of data manipulation are allowed on referenced values and how these actions affect dependent values. The rules associated with referential integrity are:

- Restrict: Disallows the update or deletion of referenced data.
- Set to Null: When referenced data is updated or deleted, all associated dependent data is set to NULL.
- Set to Default: When referenced data is updated or deleted, all associated dependent data is set to a default value.
- Cascade: When referenced data is updated, all associated dependent data is correspondingly updated. When a referenced row is deleted, all associated dependent rows are deleted.
- No Action: Disallows the update or deletion of referenced data. This differs from Restrict in that it is checked at the end of the statement, or at the end of the transaction if the constraint is deferred.

	6. Complex Integrity Checking Complex integrity checking is a user-defined rule for a column (or set of columns) that allows or disallows inserts, updates, or deletes of a row based on the value it contains for the column (or set of columns). **7. Default Value** Default value (if any)—The value an attribute instance will have if a value is not entered. **8. Field Overstuffing** a common database design problem is overstuffing columns, which is actually a normalization issue. Sometimes a single column is used for convenience to store what should be two or three columns. Such design flaws are introduced when the DBA does not analyze the data for patterns and relationships. An example of overstuffing would be storing a person's name in a single column instead of capturing first name, middle initial, and last name as individual columns. **9. Field Overloading** Overloading is the process of storing multiple facts in the same attribute. Sometimes this results from using the attribute to mean one thing for a specific type of row and another thing for a different type of row. The meaning of the attribute thus depends on the value in another attribute. Another convention is to use bits “over” another field to mean something else. Mainframe packed-decimal fields are frequently targets for overloading since they have many spare bits available. Also, numeric fields that are always expected to be positive have the sign bit available to use for something else. These practices were common on older mainframe applications and are often difficult to spot. It’s important to examine actual data values in each attribute to identify these problems. Another form of overloading is to use a single text field and load it up with keywords that represent different facts. Field overstuffing and overloading should be avoided.
Document Source Reference #	

Compliance Sources			
Name		*Website*	
Contact Information			
Name		*Website*	
Contact Information			

KEYWORDS	
List Keywords	Data element definition, overloading, overstuffing, data integrity, data quality, data type, data domain, null, key values, common values, unique values, referential integrity, default values, complex integrity

COMPONENT CLASSIFICATION				
Provide the Classification	☐ *Emerging*	☒ *Current*	☐ *Twilight*	☐ *Sunset*
Sunset Date				

COMPONENT SUB-CLASSIFICATION		
Sub-Classification	Date	Additional Sub-Classification Information
☐ *Technology Watch*		
☐ *Variance*		
☐ *Conditional Use*		

Rationale for Component Classification	
Document the Rationale for Component Classification	

Migration Strategy	
Document the Migration Strategy	

Impact Position Statement	
Document the Position Statement on Impact	

CURRENT STATUS				
Provide the Current Status	☐ *In Development*	☐ *Under Review*	☒ *Approved*	☐ *Rejected*

AUDIT TRAIL			
Creation Date	04-06-05	*Date Approved / Rejected*	6/14/05
Reason for Rejection			
Last Date Reviewed		*Last Date Updated*	
Reason for Update			